6. QR factorization

- QR factorization
- QR via Gram-Schmidt
- modified Gram-Schmidt
- pivoted QR factorization
- Householder algorithm

QR factorization

A is an $m \times n$ matrix with linearly independent columns ($m \geq n$, $\operatorname{rank}(A) = n$)

QR factorization (*reduced* or *thin* QR factorization)

$$A = QR$$

- R is $n \times n$, upper triangular, with nonzero diagonal elements (invertible)
- Q is $m \times n$ with orthonormal columns ($Q^T Q = I$)
- several algorithms, including Gram-Schmidt (in MATLAB: `[Q,R]=qr(A,0)`)

Full QR factorization (*QR decomposition*)

$$A = \begin{bmatrix} Q & Q_0 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix}$$

- same Q, R as before
- $\begin{bmatrix} Q & Q_0 \end{bmatrix}$ is $m \times m$ and orthogonal; Q_0 has size $m \times (m - n)$
- several algorithms, including Householder (in MATLAB: `[Q,R]=qr(A)`)

Pseudo-inverse via QR factorization

pseudo-inverse of A with linearly independent columns with $A = QR$ is

$$\begin{aligned} A^\dagger &= (A^TA)^{-1}A^T \\ &= \left((QR)^T(QR)\right)^{-1}(QR)^T \\ &= (R^TQ^TQR)^{-1}R^TQ^T \\ &= (R^TR)^{-1}R^TQ^T \quad (Q^TQ = I) \\ &= R^{-1}R^{-T}R^TQ^T \quad (R \text{ is nonsingular}) \\ &= R^{-1}Q^T \end{aligned}$$

- for square nonsingular A this is the inverse: $A^{-1} = (QR)^{-1} = R^{-1}Q^T$
- pseudo-inverse of A with linearly independent rows with $A^T = \hat{Q}\hat{R}$ is

$$A^\dagger = A^T(AA^T)^{-1} = \hat{Q}\hat{R}^{-T}$$

Range and QR factorization

suppose A has linearly independent columns with QR factorization $A = QR$

- Q has the same range as A:

$$\begin{aligned} y \in \mathrm{range}(A) &\iff y = Ax \text{ for some } x \\ &\iff y = QRx \text{ for some } x \\ &\iff y = Qz \text{ for some } z \\ &\iff y \in \mathrm{range}(Q) \end{aligned}$$

- columns of Q are orthonormal *basis* for $\mathrm{range}(A)$:

  they are linearly independent and span$(q_1, \ldots, q_n) = \mathrm{range}(A)$

Outline

- QR factorization
- **QR via Gram-Schmidt**
- modified Gram-Schmidt
- pivoted QR factorization
- Householder algorithm

Matrix form of Gram-Schmidt

let A be an $m \times n$ matrix with linearly independent columns

- running Gram-Schmidt on A produces orthonormal vectors $q_1, \ldots, q_n$
- we know from Gram-Schmidt algorithm that

$$\begin{aligned} a_k &= (q_1^T a_k) q_1 + \cdots + (q_{k-1}^T a_k) q_{k-1} + \|\tilde{q}_k\| q_k \\ &= R_{1k} q_1 + \cdots + R_{k-1,k} q_{k-1} + R_{kk} q_k \end{aligned}$$

  where $R_{ij} = q_i^T a_j$ and $R_{ii} = \|\tilde{q}_i\| > 0$

- expressing this for each $k = 1, \ldots, n$,

$$\begin{aligned} a_1 &= R_{11} q_1 \\ a_2 &= R_{12} q_1 + R_{22} q_2 \\ &\vdots \\ a_n &= R_{1n} q_1 + \cdots + R_{nn} q_n \end{aligned} \qquad A = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & \ldots & R_{1n} \\ 0 & R_{22} & \ldots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & R_{nn} \end{bmatrix}$$

QR factorization via Gram-Schmidt

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$
set $q_1 = a_1/\|a_1\|$ and $R_{11} = \|a_1\|$

for $k = 2, \ldots, n$

1. $\tilde{q}_k = a_k$
2. **for** $j = 1, \ldots, k-1$
   $$R_{jk} = q_j^T a_k$$
   $$\tilde{q}_k = \tilde{q}_k - R_{jk} q_j$$
3. set
   $$R_{kk} = \|\tilde{q}_k\|$$
   $$q_k = \tilde{q}_k / R_{kk}$$

- R is generated column by column
- **complexity:** $2mn^2$ flops

Example

$$A = [a_1 \;\; a_2 \;\; a_3] = \begin{bmatrix} -1 & -1 & 1 \\ 1 & 3 & 3 \\ -1 & -1 & 5 \\ 1 & 3 & 7 \end{bmatrix}$$

- $k = 1$,

$$q_1 = a_1/\|a_1\| = (-1/2, 1/2, -1/2, 1/2), \quad R_{11} = \|a_1\| = 2$$

- $k = 2$, we have $R_{12} = q_1^T a_2 = 4$, and

$$\tilde{q}_2 = a_2 - (q_1^T a_2) q_1 = (1, 1, 1, 1)$$

normalizing, we get

$$R_{22} = \|\tilde{q}_2\| = 2, \quad q_2 = \tilde{q}_2/R_{22} = (1/2, 1/2, 1/2, 1/2)$$

- $k = 3$; we have $R_{13} = q_1^T a_3 = 2$ and $R_{23} = q_2^T a_3 = 8$, so

$$\tilde{q}_3 = a_3 - (q_1^T a_3) q_1 - (q_2^T a_3) q_2 = (-2, -2, 2, 2)$$

  normalizing, we get

$$R_{33} = \|\tilde{q}_3\| = 4, \quad q_3 = \tilde{q}_3 / R_{33} = (-1/2, -1/2, 1/2, 1/2)$$

therefore,

$$\begin{aligned}
\begin{bmatrix} -1 & -1 & 1 \\ 1 & 3 & 3 \\ -1 & -1 & 5 \\ 1 & 3 & 7 \end{bmatrix} &= \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & R_{33} \end{bmatrix} \\
&= \begin{bmatrix} -1/2 & 1/2 & -1/2 \\ 1/2 & 1/2 & -1/2 \\ -1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} 2 & 4 & 2 \\ 0 & 2 & 8 \\ 0 & 0 & 4 \end{bmatrix}
\end{aligned}$$

Numerical instability of G-S

consider the following MATLAB implementation of the G-S algorithm

```
[m, n] = size(A);
Q = zeros(m,n);
R = zeros(n,n);
for k = 1:n
R(1:k-1,k) = Q(:,1:k-1)' * A(:,k);
qtilde = A(:,k) - Q(:,1:k-1) * R(1:k-1,k);
R(k,k) = norm(qtilde);
Q(:,k) = qtilde / R(k,k);
end;
```

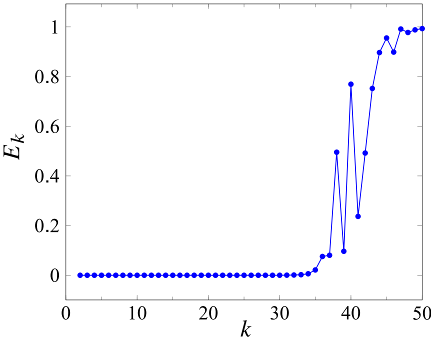
- we apply this to a square matrix A of size $m = n = 50$
- A is constructed as $A = USV$ with U, V orthogonal, S diagonal with

$$S_{ii} = 10^{-10(i-1)/(n-1)}, \quad i = 1, \ldots, n$$

Numerical instability of G-S

plot shows deviation from orthogonality between q_k and previous columns

$$E_k = \max_{1 \le i < k} |q_i^T q_k|, \quad k = 2, \ldots, n$$



loss of orthogonality is due to rounding error

Outline

- QR factorization
- QR via Gram-Schmidt
- **modified Gram-Schmidt**
- pivoted QR factorization
- Householder algorithm

Modified Gram-Schmidt algorithm

a variation of the classical Gram-Schmidt algorithm for QR factorization

$$\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1n} \\ 0 & R_{22} & \cdots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_{nn} \end{bmatrix}$$

of a matrix with linearly independent columns

- has better numerical properties than classical Gram-Schmidt algorithm
- computes Q column by column, R row by row

Modified Gram-Schmidt algorithm

after k steps ($k = 1, \ldots, n$), the algorithm has computed a partial QR factorization

$$
\begin{aligned}
A &= [a_1 \ \cdots \ a_k \mid a_{k+1} \ \cdots \ a_n] \\
&= \left[q_1 \ \cdots \ q_k \mid \tilde{Q}_k\right] \left[\begin{array}{ccc|ccc} R_{11} & \cdots & R_{1k} & R_{1,k+1} & \cdots & R_{1n} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & R_{kk} & R_{k,k+1} & \cdots & R_{kn} \\ \hline & 0 & & & I & \end{array}\right]
\end{aligned}
$$

- $q_1, \ldots, q_k$ are orthonormal vectors; $R_{11}, \ldots, R_{kk}$ are positive
- columns of $\tilde{Q}_k$ are residual of $a_{k+1}, \ldots, a_n$ after projection on span$(q_1, \ldots, q_k)$
- the factorization starts with $\tilde{Q}_0 = A$ and is complete when $k = n$
- in step k, we compute

$$q_k, \ \ R_{kk}, \ \ R_{k,k+1}, \ldots, R_{kn}, \ \ \tilde{Q}_k$$

  - compute q_k then orthogonalize each of the remaining vectors against it
  - generating R by rows rather than by columns

Modified Gram-Schmidt update

at step k we compute $q_k, R_{kk}, R_{k,(k+1):n}$, and $\tilde{Q}_k$ from

$$\tilde{Q}_{k-1} = \begin{bmatrix} q_k & \tilde{Q}_k \end{bmatrix} \begin{bmatrix} R_{kk} & R_{k,(k+1):n} \\ 0 & I \end{bmatrix}$$

partition $\tilde{Q}_{k-1}$ as $\tilde{Q}_{k-1} = [\tilde{q}_k \;\; B]$ with $\tilde{q}_k$ the first column and B of size $m \times (n-k)$:

$$\tilde{q}_k = q_k R_{kk}, \quad B = q_k R_{k,(k+1):n} + \tilde{Q}_k$$

- from the first equation, and the required properties $\|q_k\| = 1$ and $R_{kk} > 0$:

$$R_{kk} = \|\tilde{q}_k\|, \quad q_k = \frac{1}{R_{kk}} \tilde{q}_k$$

- from the second equation, and the requirement that $q_k^T \tilde{Q}_k = 0$:

$$R_{k,(k+1):n} = q_k^T B, \quad \tilde{Q}_k = (I - q_k q_k^T) B = B - q_k R_{k,(k+1):n}$$

Summary: modified Gram-Schmidt algorithm

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$
set $\tilde{Q}_0 = A$
for $k = 1$ to n,

1. compute $R_{kk} = \|\tilde{q}_k\|$ and $q_k = (1/R_{kk})\tilde{q}_k$ where $\tilde{q}_k$ is the first column of $\tilde{Q}_{k-1}$
2. compute

$$[R_{k,k+1} \; \cdots \; R_{kn}] = q_k^T B, \quad \tilde{Q}_k = B - q_k [R_{k,k+1} \; \cdots \; R_{kn}]$$

   where B is $\tilde{Q}_{k-1}$ with the first column removed

complexity: $2mn^2$ flops

MATLAB implementation (`Q(:,k:n)` used to store $\tilde{Q}_{k-1}$)

```
Q = A; R = zeros(n,n);
for k = 1:n
R(k,k) = norm(Q(:,k));
Q(:,k) = Q(:,k) / R(k,k);
R(k,k+1:n) = Q(:,k)' * Q(:,k+1:n);
Q(:,k+1:n) = Q(:,k+1:n) - Q(:,k) * R(k,k+1:n);
end;
```

Example

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 1 \\ 1 & 3 & 3 \\ -1 & -1 & 5 \\ 1 & 3 & 7 \end{bmatrix}$$

Step 1: first column of Q, first row of R

$$\begin{aligned} \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} &= \left[\begin{array}{c|cc} -1/2 & 1 & 2 \\ 1/2 & 1 & 2 \\ -1/2 & 1 & 6 \\ 1/2 & 1 & 6 \end{array}\right] \left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \\ &= \left[\begin{array}{c|c} q_1 & \tilde{Q}_2 \end{array}\right] \left[\begin{array}{c|c} R_{11} & R_{1,2:3} \\ \hline 0 & I \end{array}\right] \end{aligned}$$

Example

Step 2: second column of Q, second row of R

$$\left[\begin{array}{ccc} a_1 & a_2 & a_3 \end{array}\right] = \left[\begin{array}{cc|c} -1/2 & 1/2 & -2 \\ 1/2 & 1/2 & -2 \\ -1/2 & 1/2 & 2 \\ 1/2 & 1/2 & 2 \end{array}\right] \left[\begin{array}{cc|c} 2 & 4 & 2 \\ 0 & 2 & 8 \\ \hline 0 & 0 & 1 \end{array}\right]$$

$$= \left[\begin{array}{cc|c} q_1 & q_2 & \tilde{Q}_3 \end{array}\right] \left[\begin{array}{cc|c} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ \hline 0 & 0 & 1 \end{array}\right]$$

Step 3: third column of Q, third row of R

$$\left[\begin{array}{ccc} a_1 & a_2 & a_3 \end{array}\right] = \left[\begin{array}{ccc} -1/2 & 1/2 & -1/2 \\ 1/2 & 1/2 & -1/2 \\ -1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 \end{array}\right] \left[\begin{array}{ccc} 2 & 4 & 2 \\ 0 & 2 & 8 \\ 0 & 0 & 4 \end{array}\right]$$

$$= \left[\begin{array}{ccc} q_1 & q_2 & q_3 \end{array}\right] \left[\begin{array}{ccc} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & R_{33} \end{array}\right]$$

Outline

- QR factorization
- QR via Gram-Schmidt
- modified Gram-Schmidt
- **pivoted QR factorization**
- Householder algorithm

QR factorization with column pivoting

A is an $m \times n$ matrix (may be wide or have linearly dependent columns)

QR factorization with column pivoting (column reordering)

$$A = QRP^T$$

- Q is $m \times r$ with orthonormal columns
- R is $r \times n$, leading $r \times r$ submatrix is upper triangular with positive diagonal:

$$R = \left[\begin{array}{c|c} R_1 & R_2 \end{array}\right] = \left[\begin{array}{cccc|ccc} R_{11} & R_{12} & \cdots & R_{1r} & R_{1,r+1} & \cdots & R_{1n} \\ 0 & R_{22} & \cdots & R_{2r} & R_{2,r+1} & \cdots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & R_{rr} & R_{r,r+1} & \cdots & R_{rn} \end{array}\right]$$

- can be chosen to satisfy $R_{11} \geq R_{22} \geq \cdots \geq R_{rr} > 0$
- P is an $n \times n$ permutation matrix
- r is the rank of A: gives full rank factorization $A = BC$ with $B = Q$, $C = RP^T$

Interpretation

- columns of $AP = QR$ are the columns of A in a different order
- the columns are divided in two groups:

$$AP = [\hat{A}_1 \;\; \hat{A}_2] = Q[R_1 \;\; R_2], \quad \hat{A}_1 \text{ is } m \times r, \; R_1 \text{ is } r \times r$$

- $\hat{A}_1 = QR_1$ is $m \times r$ with linearly independent columns:

$$\hat{A}_1 x = QR_1 x = 0 \quad \Longrightarrow \quad R_1^{-1} Q^T \hat{A}_1 x = x = 0$$

- $\hat{A}_2 = QR_2$ is $m \times (n - r)$: columns are linear combinations of columns of $\hat{A}_1$

$$\hat{A}_2 = QR_2 = \hat{A}_1 R_1^{-1} R_2$$

the factorization provides two useful bases for $\mathrm{range}(A)$

- columns of Q are an orthonormal basis
- columns of $\hat{A}_1$ are a basis selected from the columns of A

Dimension of nullspace

if A is $m \times n$ then

$$\dim(\mathrm{null}(A)) = n - \mathrm{rank}(A)$$

- $\dim(\mathrm{null}(A))$ is known as the *nullity* of the matrix
- we show this by constructing a basis containing $n - \mathrm{rank}(A)$ vectors

Basis for nullspace: a basis for the nullspace of A is given by the columns of

$$P \begin{bmatrix} -R_1^{-1} R_2 \\ I \end{bmatrix}$$

where P, R_1, R_2 are the matrices in the pivoted QR factorization

$$AP = Q \begin{bmatrix} R_1 & R_2 \end{bmatrix}$$

- P is a $n \times n$ permutation matrix
- Q is $m \times r$ with orthonormal columns, where $r = \mathrm{rank}(A)$
- R_1 is $r \times r$ upper triangular and nonsingular, R_2 is $r \times (n - r)$

Proof

- x is in the nullspace of A if and only if $y = P^T x$ is in the nullspace of AP
- $y = (y_1, y_2)$ is in the nullspace of AP if and only if

$$
\begin{aligned}
APy = 0 &\Longleftrightarrow Q \begin{bmatrix} R_1 & R_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 0 \\
&\Longleftrightarrow \begin{bmatrix} R_1 & R_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 0 \qquad (Q \text{ has orthonormal columns}) \\
&\Longleftrightarrow \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} -R_1^{-1} R_2 \\ I \end{bmatrix} y_2 \qquad (R_1 \text{ nonsingular})
\end{aligned}
$$

- therefore, x is in the nullspace of A if and only if it is in the range of

$$P \begin{bmatrix} -R_1^{-1} R_2 \\ I \end{bmatrix}$$

- the columns of this matrix are linearly independent, so they are a basis for

$$\mathrm{range}\left(P \begin{bmatrix} -R_1^{-1} R_2 \\ I \end{bmatrix}\right) = \mathrm{null}(A)$$

Modified Gram-Schmidt algorithm with pivoting

with minor changes the modified GS algorithm computes the pivoted factorization

$$AP = \begin{bmatrix} q_1 & q_2 & \cdots & q_r \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1r} & R_{1,r+1} & \cdots & R_{1n} \\ 0 & R_{22} & \cdots & R_{1r} & R_{1,r+1} & \cdots & R_{1n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & R_{rr} & R_{r,r+1} & \cdots & R_{rn} \end{bmatrix}$$

- partial factorization after k steps

$$AP_k = \begin{bmatrix} q_1 \cdots q_k \mid \tilde{Q}_k \end{bmatrix} \left[\begin{array}{ccc|ccc} R_{11} & \cdots & R_{1k} & R_{1,k+1} & \cdots & R_{1n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & R_{kk} & R_{k,k+1} & \cdots & R_{kn} \\ \hline & 0 & & & I & \end{array}\right]$$

- if $\tilde{Q}_k = 0$, the factorization is complete ($r = k, P = P_k$)
- algorithm starts with $P_0 = I$ and $\tilde{Q}_0 = A$
- before step k, we reorder columns of $\tilde{Q}_{k-1}$ to place its largest column first
- this requires reordering columns $k, \ldots, n$ of R, and modifying P_{k-1}

Example

$$A = \left[\begin{array}{cccc} a_1 & a_2 & a_3 & a_4 \end{array}\right] = \left[\begin{array}{cccc} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & -1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & -1 & -1 \end{array}\right]$$

Step 1

- a_2 and a_4 have the largest norms; we move a_2 to the first position
- find first column of Q, first row of R

$$\left[\begin{array}{cccc} a_2 & a_1 & a_3 & a_4 \end{array}\right] = \left[\begin{array}{c|ccc} 1/2 & 1/2 & 0 & 1 \\ 1/2 & -1/2 & 1 & -1 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & -1/2 & -1 & -1 \end{array}\right] \left[\begin{array}{c|ccc} 2 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

$$= \left[q_1 \mid \tilde{Q}_1\right] \left[\begin{array}{c|c} R_{11} & R_{1,2:4} \\ \hline 0 & I \end{array}\right]$$

Example

Step 2

- move column 3 of $\tilde{Q}_1$ to first position in $\tilde{Q}_1$

$$\left[\begin{array}{cccc} a_2 & a_4 & a_1 & a_3 \end{array}\right] = \left[\begin{array}{c|ccc} 1/2 & 1 & 1/2 & 0 \\ 1/2 & -1 & -1/2 & 1 \\ 1/2 & 1 & 1/2 & 0 \\ 1/2 & -1 & -1/2 & -1 \end{array}\right] \left[\begin{array}{c|ccc} 2 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

- find second column of Q, second row of R

$$\begin{aligned} \left[\begin{array}{cccc} a_2 & a_4 & a_1 & a_3 \end{array}\right] &= \left[\begin{array}{cc|cc} 1/2 & 1/2 & 0 & 0 \\ 1/2 & -1/2 & 0 & 1 \\ 1/2 & 1/2 & 0 & 0 \\ 1/2 & -1/2 & 0 & -1 \end{array}\right] \left[\begin{array}{cc|cc} 2 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \\ &= \left[\begin{array}{cc|c} q_1 & q_2 & \tilde{Q}_2 \end{array}\right] \left[\begin{array}{cc|c} R_{11} & R_{12} & R_{1,3:4} \\ 0 & R_{22} & R_{2,3:4} \\ \hline 0 & 0 & I \end{array}\right] \end{aligned}$$

Example

Step 3

- move column 2 of $\tilde{Q}_2$ to first position in $\tilde{Q}_2$

$$\left[\begin{array}{cccc} a_2 & a_4 & a_3 & a_1 \end{array}\right] = \left[\begin{array}{cc|cc} 1/2 & 1/2 & 0 & 0 \\ 1/2 & -1/2 & 1 & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 1/2 & -1/2 & -1 & 0 \end{array}\right] \left[\begin{array}{cc|cc} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

- find third column of Q, third row of R

$$\begin{aligned} \left[\begin{array}{cccc} a_2 & a_4 & a_3 & a_1 \end{array}\right] &= \left[\begin{array}{ccc|c} 1/2 & 1/2 & 0 & 0 \\ 1/2 & -1/2 & 1/\sqrt{2} & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 1/2 & -1/2 & -1/\sqrt{2} & 0 \end{array}\right] \left[\begin{array}{ccc|c} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & \sqrt{2} & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \\ &= \left[\begin{array}{ccc|c} q_1 & q_2 & q_3 & \tilde{Q}_3 \end{array}\right] \left[\begin{array}{ccc|c} R_{11} & R_{12} & R_{13} & R_{14} \\ 0 & R_{22} & R_{23} & R_{24} \\ 0 & 0 & R_{33} & R_{34} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \end{aligned}$$

Example

Result: since $\tilde{Q}_3$ is zero, the algorithm terminates with the factorization

$$\begin{bmatrix} a_2 & a_4 & a_3 & a_1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & -1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & -1 & -1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & -1/2 & 1/\sqrt{2} \\ 1/2 & 1/2 & 0 \\ 1/2 & -1/2 & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & \sqrt{2} & 0 \end{bmatrix}$$

Full pivoted QR factorization

any $A \in \mathbb{R}^{m\times n}$ admits the full pivoted QR factorization:

$$A = \begin{bmatrix} Q & Q_0 \end{bmatrix} \begin{bmatrix} R_1 & R_2 \\ 0 & 0 \end{bmatrix} P^T$$

- $Q^TQ = I$, $R_1 \in \mathbb{R}^{r\times r}$ is upper triangular and invertible
- $R_2 \in \mathbb{R}^{r\times(n-r)}$, and $P \in \mathbb{R}^{n\times n}$ is a permutation matrix
- $[Q \;\; Q_0]$ is an $m \times m$ orthogonal matrix
- columns of Q_0 are an orthonormal basis for $\mathrm{null}(A^T)$:

$$\mathrm{range}(Q_0) = \mathrm{range}(A)^\perp = \mathrm{null}(A^T)$$

this follows from

$$A^T = P[R^T \;\; 0] \begin{bmatrix} Q^T \\ Q_0^T \end{bmatrix}$$

and

$$A^Tz = PR^TQ^Tz = 0 \quad \Longleftrightarrow \quad Q^Tz = 0 \quad \Longleftrightarrow \quad z \in \mathrm{range}(Q_0)$$

Outline

- QR factorization
- QR via Gram-Schmidt
- modified Gram-Schmidt
- pivoted QR factorization
- **Householder algorithm**

Householder algorithm

- the most widely used algorithm for QR factorization (`qr` in MATLAB and Julia)
- less sensitive to rounding error than (modified) Gram-Schmidt algorithm
- computes a "full" QR factorization (QR decomposition)

$$A = \begin{bmatrix} Q & Q_0 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad [Q \;\; Q_0] \text{ orthogonal}$$

- can be modified to compute pivoted QR factorization:

$$A = \begin{bmatrix} Q & Q_0 \end{bmatrix} \begin{bmatrix} R_1 & R_2 \\ 0 & 0 \end{bmatrix} P^T = Q \underbrace{[R_1 \;\; R_2]}_{=R} P^T$$

  where $R_1 \in \mathbb{R}^{r \times r}$ is upper triangular and invertible ($r = \mathrm{rank}(A)$)

- the full Q-factor is constructed as a product of orthogonal matrices

$$[Q \;\; Q_0] = H_1 H_2 \cdots H_n$$

  each H_i is an $m \times m$ symmetric and orthogonal

Reflector

Reflector: an *elementary reflector* is a matrix of the form

$$H = I - 2vv^T \quad \text{with } v \text{ a unit-norm vector } \|v\| = 1$$

Properties

- a reflector matrix is symmetric, and orthogonal

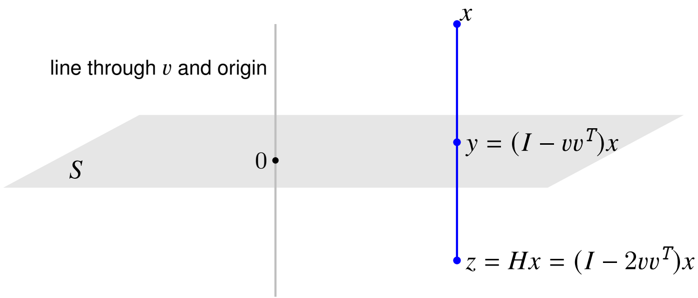
$$H^T H = (I - 2vv^T)(I - 2vv^T) = I - 4vv^T + 4vv^T vv^T = I$$

- reflection of v: $Hv = -v$
- matrix-vector product Hx can be computed efficiently as

$$Hx = x - 2(v^T x)v$$

  complexity is $4p$ flops if v and x have length p

Geometrical interpretation of reflector



- $S = \{u \mid v^T u = 0\}$ is the (hyper-)plane of vectors orthogonal to v
- if $\|v\| = 1$, the projection of x on S is given by

$$y = (I - vv^T)x$$

- reflection of x through the hyperplane is given by product with reflector:

$$z = y + (y - x) = (I - 2vv^T)x$$

Reflection to multiple of first unit vector

given nonzero p-vector $y = (y_1, y_2, \ldots, y_p)$, define

$$w = \begin{bmatrix} y_1 + \operatorname{sign}(y_1)\|y\| \\ y_2 \\ \vdots \\ y_p \end{bmatrix}, \quad v = \frac{1}{\|w\|} w$$

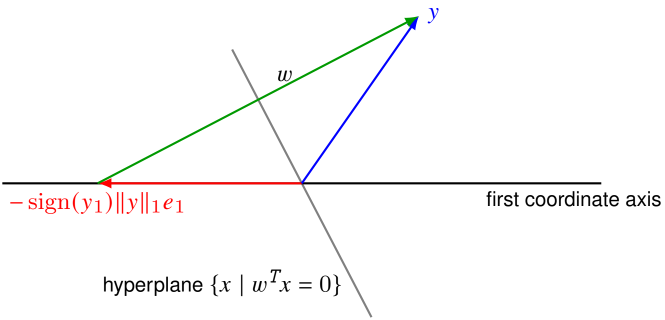
- we define $\operatorname{sign}(0) = 1$
- vector w satisfies

$$\|w\|^2 = 2(w^T y) = 2\|y\|(\|y\| + |y_1|)$$

- reflector $H = I - 2vv^T$ maps y to multiple of $e_1 = (1, 0, \ldots, 0)$:

$$Hy = y - \frac{2(w^T y)}{\|w\|^2} w = y - w = -\operatorname{sign}(y_1)\|y\| e_1$$

Geometry



the reflection through the hyperplane $\{x \mid w^T x = 0\}$ with normal vector

$$w = y + \operatorname{sign}(y_1)\|y\|e_1$$

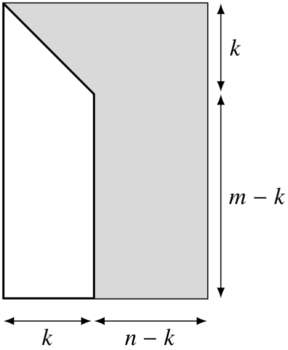
maps y to the vector $-\operatorname{sign}(y_1)\|y\|e_1$

Householder triangularization

- computes reflectors $H_1, \ldots, H_n$ that reduce A to triangular form:

$$H_n H_{n-1} \cdots H_1 A = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

- after step k, the matrix $H_k H_{k-1} \cdots H_1 A$ has the following structure:



(elements in positions i, j for $i > j$ and $j \leq k$ are zero)

Householder algorithm

---

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$

for $k = 1, 2, \ldots, n$

1. define $y = A_{k:m,k}$ and compute $(m-k+1)$-vector v_k:

$$w = y + \operatorname{sign}(y_1)\|y\|e_1, \quad v_k = \frac{1}{\|w\|}w$$

2. multiply $A_{k:m,k:n}$ with reflector $I - 2v_k v_k^T$:

$$A_{k:m,k:n} := A_{k:m,k:n} - 2v_k(v_k^T A_{k:m,k:n})$$

- algorithm overwrites A with $\begin{bmatrix} R \\ 0 \end{bmatrix}$
- **complexity:** $2mn^2 - \frac{2}{3}n^3$ flops (we take $2mn^2$ for the complexity)

Remarks

- step 2 is equivalent to multiplying A with $m \times m$ reflector

$$H_k = \begin{bmatrix} I & 0 \\ 0 & I - 2v_k v_k^T \end{bmatrix} = I - 2 \begin{bmatrix} 0 \\ v_k \end{bmatrix} \begin{bmatrix} 0 \\ v_k \end{bmatrix}^T$$

- algorithm returns the vectors $v_1, \ldots, v_n$, with v_k of length $m - k + 1$

Q-factor

$$\begin{bmatrix} Q & Q_0 \end{bmatrix} = H_1 H_2 \cdots H_n$$

- usually there is no need to compute the matrix $[Q \;\; Q_0]$ explicitly
- the vectors $v_1, \ldots, v_n$ are an economical representation of $[Q \;\; Q_0]$
- products with $[Q \;\; Q_0]$ or its transpose can be computed as

$$\begin{bmatrix} Q & Q_0 \end{bmatrix} x = H_1 H_2 \cdots H_n x$$
$$\begin{bmatrix} Q & Q_0 \end{bmatrix}^T y = H_n H_{n-1} \cdots H_1 y$$

Example

$$A = \begin{bmatrix} -1 & -1 & 1 \\ 1 & 3 & 3 \\ -1 & -1 & 5 \\ 1 & 3 & 7 \end{bmatrix} = H_1 H_2 H_3 \begin{bmatrix} R \\ 0 \end{bmatrix}$$

we compute reflectors H_1, H_2, H_3 that triangularize A:

$$H_3 H_2 H_1 A = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & R_{33} \\ 0 & 0 & 0 \end{bmatrix}$$

First column of R

- compute reflector that maps first column of A to multiple of e_1:

$$y = \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}, \quad w = y - \|y\|e_1 = \begin{bmatrix} -3 \\ 1 \\ -1 \\ 1 \end{bmatrix}, \quad v_1 = \frac{1}{\|w\|}w = \frac{1}{2\sqrt{3}} \begin{bmatrix} -3 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

- overwrite A with product of $I - 2v_1v_1^T$ and A

$$A := (I - 2v_1v_1^T)A = \begin{bmatrix} 2 & 4 & 2 \\ 0 & 4/3 & 8/3 \\ 0 & 2/3 & 16/3 \\ 0 & 4/3 & 20/3 \end{bmatrix}$$

Second column of R

- compute reflector that maps $A_{2:4,2}$ to multiple of e_1:

$$y = \begin{bmatrix} 4/3 \\ 2/3 \\ 4/3 \end{bmatrix}, \quad w = y + \|y\| e_1 = \begin{bmatrix} 10/3 \\ 2/3 \\ 4/3 \end{bmatrix}, \quad v_2 = \frac{1}{\|w\|} w = \frac{1}{\sqrt{30}} \begin{bmatrix} 5 \\ 1 \\ 2 \end{bmatrix}$$

- overwrite $A_{2:4,2:3}$ with product of $I - 2v_2 v_2^T$ and $A_{2:4,2:3}$:

$$A := \begin{bmatrix} 1 & 0 \\ 0 & I - 2v_2 v_2^T \end{bmatrix} A = \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & 16/5 \\ 0 & 0 & 12/5 \end{bmatrix}$$

Third column of R

- compute reflector that maps $A_{3:4,3}$ to multiple of e_1:

$$y = \begin{bmatrix} 16/5 \\ 12/5 \end{bmatrix}, \quad w = y + \|y\|e_1 = \begin{bmatrix} 36/5 \\ 12/5 \end{bmatrix}, \quad v_3 = \frac{1}{\|w\|}w = \frac{1}{\sqrt{10}} \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

- overwrite $A_{3:4,3}$ with product of $I - 2v_3v_3^T$ and $A_{3:4,3}$:

$$A := \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} A = \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & -4 \\ 0 & 0 & 0 \end{bmatrix}$$

Final result

$$
\begin{aligned}
H_3H_2H_1A &= \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & I - 2v_2v_2^T \end{bmatrix} (I - 2v_1v_1^T)A \\
&= \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & I - 2v_2v_2^T \end{bmatrix} \begin{bmatrix} 2 & 4 & 2 \\ 0 & 4/3 & 8/3 \\ 0 & 2/3 & 16/3 \\ 0 & 4/3 & 20/3 \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & 16/5 \\ 0 & 0 & 12/5 \end{bmatrix} \\
&= \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & -4 \\ 0 & 0 & 0 \end{bmatrix}
\end{aligned}
$$

References and further readings

- S. Boyd and L. Vandenberghe. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares.* Cambridge University Press, 2018.
- L. Vandenberghe, *EE133A Lecture Notes,* University of California, Los Angeles.
- L. Vandenberghe. *EE133B Lecture Notes*, University of California, Los Angeles.
- M. T. Heath. *Scientific Computing: An Introductory Survey* (revised second edition). Society for Industrial and Applied Mathematics, 2018.
- U. M. Ascher. *A First Course on Numerical Methods*. Society for Industrial and Applied Mathematics, 2011.